

## BASIC SORTING SOLUTIONS

Solution (a): Bubble Sort

```
public static void bubbleSortDescending(int arr[]) {  
    for(int turn=0; turn<arr.length-1; turn++) {  
        for(int j=0; j<arr.length-1-turn; j++) {  
            if(arr[j] < arr[j+1]) {  
                //swap  
                int temp = arr[j];  
                arr[j] = arr[j+1];  
                arr[j+1] = temp;  
            }  
        }  
    }  
}
```

Solution (b): Selection Sort

```
public static void selectionSortDescending(int arr[]) {  
    for(int turn=0; turn<arr.length; turn++) {  
        int minPos = turn;  
        for(int j=turn+1; j<arr.length; j++) {  
            if(arr[minPos] < arr[j]) {  
                minPos = j;  
            }  
        }  
  
        //swap  
        int temp = arr[turn];  
        arr[turn] = arr[minPos];  
        arr[minPos] = temp;  
    }  
}
```

Solution (c) : Insertion Sort

```
public static void insertionSortDescending(int arr[]) {
```

prakashtiwari06@gmail.com

```
for(int i=1; i<arr.length; i++) {  
    int curr = arr[i];  
    int prev = i-1;  
    //to find the index where curr is to be inserted  
    while(prev >= 0 && arr[prev] < curr) {  
        arr[prev+1] = arr[prev];  
        prev--;  
    }  
    arr[prev+1] = curr;  
}  
}
```

#### Solution (d): Counting Sort

```
public static void countingSortDescending(int arr[]) {  
    int largest = Integer.MIN_VALUE;  
    for(int i=0; i<arr.length; i++) {  
        largest = Math.max(largest, arr[i]);  
    }  
  
    int count[] = new int[largest+1];  
    for(int i=0; i<arr.length; i++) {  
        count[arr[i]]++;  
    }  
    int j = 0;  
    for(int i=count.length-1; i>=0; i--) {  
        while(count[i] > 0) {  
            arr[j] = i;  
            j++;  
            count[i]--;
        }
    }
}
```

prakashtiwari06@gmail.com